LogNet: Extending Internet with a Network Aware Discovery Service

Luigi Liquori, Matteo Sereno

LogNet: Extending Internet with a Network Aware Discovery Service

[Extended abstract]

Luigi Liquori
Université Côte d'Azur
Inria Sophia Antipolis Méditerranée, France
Email: Luigi.Liquori@inria.fr

Matteo Sereno
Università di Torino, Italy
Email: Matteo.Sereno@unito.it

***Abstract*—Everything needs to change, so everything can stay the same**[1]. **Internet in recent years has become a huge set of channels for content distribution. And this has highlighted limits and inefficiencies of the current protocol suite originally designed for host-to-host communication. This paper joins the research efforts addressed by the new Internet challenges by proposing LogNet, a conservative extension of the current TCP/IP hourglass Internet architecture, that provides a new network aware Content Discovery Service.**

Contents are referred via the new notion of *HyperNames* (HN), whose rich syntax allow to specify, hosts, pki, fingerprint and a large list of *optional logical attributes* (tags) attached to the content name, such as mutable vs immutable contents, digital signatures, ownership, availability, price, etc. HyperNames are in part human-readable and in part machine-readable and only in the latter case self-certifying.

Publication and discovery of HN is achieved using the new distributed service *Content Name System* (CNS) with related protocol, whose behavior and architecture is, partly, inspired by the DNS, and whose "routing logic" uses the BGP inter domain routing information.

The core of CNS is the *HyperName Lookup Algorithm* (HLA) which "tunes" content discovery of being network aware, by exploiting the Autonomous System (AS) relationships. In particular, the HLA starts the content discovery process in the local AS (i.e., where the query starts), and in case of negative answer, propagate the query by accounting for the AS-to-AS relationships (i.e., peering, provider-to-customer, customer-to-provider). After discovered the owner(s) or the purveyor(s) of the content we are looking for, the latter can be retrieved using common transfer protocols (centralized or distributed), since the actors of this transfer are chosen in a network aware fashion (i.e., as close as possible one to each other).

I. Introduction

Information Centric Networks (ICN) is a clean-state approach to redesign the actual Internet infrastructure from a "host-centric", fully connected, paradigm to a "name-centric", loosely connected, paradigm where the focus is on named data instead of machine name hosting those data. In the last decade many proposals raised from research to capture this new paradigm: they can be grouped into two main schools of thought: (i) *Content Centric Networks* referring to the Jacobson-based vision [1], where routing is driven by fully qualified - human readable - hierarchical names, and (ii) *Data Oriented Network Architecture* (DONA) referring to a flat, human unreadable but unique name-space [2].

[1]From "The Leopard" by Giuseppe Tomasi di Lampedusa.

A. Clean-state Design vs. Evolution of the Existing Network Architecture

While it is always exciting to conceive a new network starting from new concepts and from a clean-state design, network's history teaches us that the Internet infrastructure and its protocol suite have little changed especially at the lower level of the OSI stack; this is, quite obviously, because of strong backward-compatibility needs, and because of the tremendous expansion of the Internet phenomenon.

This paper supports the "evolutive" research line. In particular, we believe that the current TCP/IP-based transport protocol could be considered as a target protocol suite to efficiently map more sophisticated protocols and services.

Only during the conference submission, a web appendix containing extra material for referees will be available on [3].

B. The network aware discovery service

Since the very beginning of this research vein, the *querelle* has been about how build a new kind of Internet based on *"Contents-as-Names"* instead of the current *"Names-as-IPs"* paradigm. The simple recipe developed in this paper is to forge a new paradigm, namely *"Contents-as-IPs"*.

This paper presents a lightweight Internet service to be implemented on the existing Internet stack, and more precisely, between the Transport and the Session layers (referring to the ISO-OSI protocol layering). We call this new service *Content Name System* (CNS) organized throughout a set of communicating CNS servers. In a nutshell, the purpose of this service is to "publish" machine-IP-addresses being the owners (or the purveyors) of some named-contents and "retrieve" that machine-IP-addresses performing a distributed search using the named-content as the database-key. In other words: the service binds, in the distributed CNS, set of IP-addresses to content-names, the latter modeled by HyperNames (HN). The CNS service stops when some or no IP-addresses are returned or when no other CNS can be delegated in the iterative call implementing the distributed data-base query.

Each CNS will naturally be equipped with a *key-value* database containing for each content-name, the set of corresponding IP ordered by local awareness: dealing with local pointers instead of the real data has many advantages. In particular, it decreases the size of the data bases in CNS servers, promotes local awareness, mobility and nomadism, reduces CNS server overhead, etc. Note that, once the CNS reply to a query giving a list of IP that are owners or purveyors of some contents, the proposed protocol terminates and the real transfer of the data proceeds using usual client-server or peer-to-peer protocols with the positive consequence that the choice of the IP "actors" has been done promoting locality awareness and respecting inter-domain routing. Note also that adding small RAM caches is a possible and suitable option.

The CNS service promotes data republication of contents whose owner/purveyors are far away in the CNS belonging to the current AS. Therefore, we could imagine as a potential use of that service, well-known peer-to-peer protocols, like BitTorrent, performing location-aware file exchange between nodes belonging to same/close ASes or, better, using some unofficial BitTorrent extensions, or even better raising the CNS servers also to the status of "location aware BitTorrent trackers".

One last remark: nowadays many P2P protocols saturate the network traffic by an "hysteric" (from the point of view of network providers) flow of network connections from one AS to another AS without taking care of the fact the Internet traffic have a *real monetary cost* caused mostly by following BGP routes and AS-to-AS relationship between ASes. This fact, probably and also the other well-known fact that mostly of the exchanged contents breaks the DMCA caused the demonization of almost all P2P protocols. This paper try to capitalize (i) the P2P experience, and (ii) the experience in studying the AS-relationships (i.e., provider-to-customer, customer-to-provider, peering relationships) [4], [5], to propose services and protocols that can be easily included in the current Internet infrastructure.

C. Content Name System (CNS)

A CNS is a hierarchical and decentralized naming system providing a new Internet service that translates content names into IP addresses needed to later retrieve the content itself. The CNS servers are distributed over the ASes, and more precisely there is *at least one* CNS server *per* AS taking care of resources registered inside the AS itself. In a nutshell, the CNS server hierarchy mimics the AS relationship hierarchy. More precisely, the CNS server, associated to a given AS, set its position in the CNS server hierarchy starting from the (private) AS business relationships. As we know, a nice approximation of those relations is snapshotted every month by CAIDA [6].

D. Names (HN)

A HN denotes a name of one content. The name is composed by an human readable part followed by an *optional* machine readable part. The content-name is enriched with a number of (optional) *parameters*, also called *logical attributes*, or *tags*, helping to identify univocally the content, ownership, its integrity, its signature, its price, availability, etc. Parameters are not enforced to be human readable. HNs are used essentially to *publish* contents on a local CNS and try to *resolve* a HN with the list of IP addresses of the owners/purveyors of the data. A HN does not enforce to denote uniquely neither the content neither the owner or the purveyor of the content but it can be achieved by using suitable logical attributes.

E. Frequently Asked Questions

This subsection tries, in a somewhat informal, to give some insight to the LogNet proposed architecture by means of **Q**-**A**.
Q: *Does LogNet can be considered as a conservative extension of the actual Internet?*
A: Yes: LogNet do not enforce to use the new CNS service: as an example, a happy user of Google web services and P2P file exchange protocols can continue to stay with it without changing their habits.
Q: *Does LogNet is based on a clean-slate design?*
A: LogNet is not based on a clean-slate design: it is small, surgical extension of the current ISO-OSI stack providing a new service allowing to publish and to discover locally aware contents by HyperNames. LogNet is simply a modest and conservative extension of the current Internet.
Q: *Does LogNet can be seen as a Google killer?*
A: No. Since LogNet enforces at least one CNS being hosted in each AS, it follows that the ability of each node to publish a content and then retrieve another is related to the HLA, taking a HN as a query input and producing, in case of a successful lookup, either a non-empty set of IPs – where the full content will be retrieved – or a failure. Nevertheless, we could say that LogNet architecture share the same Google vision/mission (namely *"to provide access to the world's information in one click/to organize the world's information and make it universally accessible and useful"*) but in a distributed fashion.
Q: *Does LogNet is* yet another *overlay network proposal?*
A: LogNet is not an overlay network: it is a *network aware content discovery service* that does not introduce another logical address space disconnected with the IP one. The LogNet architecture extend the Session Layer.
Q: *Does LogNet shares analogies with CDN, e.g. Akamai?*
A: Just a bit, CDN is an overlay network whose mission is *to help server providers to move and cache popular contents in order to leverage busy servers*: mirroring is done using an overlay network on the top of the actual Internet. LogNet is an extension of the current Internet architecture: its mission is somewhat orthogonal to CDN, namely *to help clients to locate the closest copy of the required content in the current (or closest) AS*, and the playfield locates at the Session Layer.
Q: *Does LogNet shares analogies with DNS service?*
A: Just a bit, the HLA implementation `links` share some similarities with the DNS lookup implementation of `bind`, but both works on different topologies: the DNS hierarchy is not network-aware while the CNS one follows the AS "upstream-peering-downstream" topology.
Q: *Does LogNet add "network awareness" to classic peer-to-*

peer protocols and video streaming?
A: Yes. The exact amount of network awareness needed to be carefully exploited in order to enhance scalability of P2P file exchange/streaming protocols in the current Internet.
Q: *How does LogNet get seeded?*
HNs are published in the local CNS server and eventually replicated each time a HLA on that HN resolve successfully.
Q: *By globally deploying the CNS service, should we expect a general overcrowding of the BGP routes?*
A: No because: (i) the traffic generated only by content discovery is *not exhaustive* by construction, and (ii) content replication and cache techniques can leverage CNS query lookup.
Q: *By using the IP resolved by a CNS to fetch contents from IP to IP, should we expect an overcrowding of the BGP routes?*
A: No. The CNS query explore the CNS hierarchy by performing the search flow from local first, then through customer-AS, then through peering-AS, and finally through provider-AS. If a content can be resolved, then the content will be fetched from the owner/purveyor to the demander preserving, at the best effort, the well known *no-valley routing* intra AS [4], [7].
Q: *By globally deploying the CNS service, should we expect some decrement of the global number of IP packets?*
A: Uhm, quite difficult question: the LogNet extension forces contents to be discovered and cached in the AS where they are looked for, and offer always to P2P protocols the "closest" local peers. A new kind of *"Network Aware P2P protocol"*?
Q: *The CNS protocol suite should be available at which network level?*
A: In principle, all services "above" the CNS protocol could take advantage of the discovery service, and especially services at the Application Level, such as HTTP, FTP, and SMTP, and all the P2P ones.
Q: *Why not simply use what already exists today with Web browsers/Search/URLs? Why users would they be motivated to use LogNet a discovery service over what is currently in place?*
A: (i) LogNet discovery is 100% location aware, web discovery started to be (see the Google's new location-aware search); (ii) LogNet discovery works at 4-5 OSI level, web discovery at 7, much higher; (iii) LogNet publication is done by a precise explicit primitive `publish`, web publication is just publish something in your web page and wait for a Google crawler/robot scanning. (iv) The HLA implementation is/will be is 100% open source, (v) LogNet publication can, with some restrictions, be done by a mobile peer.

II. HyperNames and Content Name System

A. HyperNames

A LogNet's *HyperName* is a name string enriched with a number of (optional) parameters helping to identify a content, and possibly its ownership, its integrity, its hosting, and its attribute-list. More formally:

Definition 2.1 (HyperName): A *HyperName* (HN) is generated by the following abstract syntax:

`[fing_princ:][fing_cont:][hosts:][tags:]cont_name`

where `cont_name` *is a mandatory, human readable, string denoting a content name,* `tags` *is the optional list of* tags *associated with a given content,* `hosts` *is the optional list of* hostnames *being the purveyors of the content,* `fing_cont` *is the optional digital signature (i.e., the cryptographic hash) of the content,* `fing_princ` *is the optional digital signature (i.e., the cryptographic hash) of the public asymmetric key of the* principal*, i.e., the owner of the content.*

In short, a HN is a human readable content name followed by some *optional* human readable tags and then by some *optional* machine readable datas. A HN *human view* can display just the content name and the tag list, leaving all the others machine readable datas to the *internal view*. By using optional prefixes in HN, we contribute to identify, search and retrieve contents and ownership in the LogNet architecture, to normalize and optimize the CNS distributed data base using *Data Deduplication techniques* [8] that can be used to improve the performance of the the HLA presented later in this section. The concept of HN as a name with "different views" was motivated by the aims of breaking the Zooko's triangle conjecture saying that no single kind of naming can achieve more than two of the following features, namely to be human readable, decentralized, and secure.

B. Content Name System

In terms of protocol stack the CNS service can be located in the IP hourglass at the same level of the DNS. In particular, the CNS provides a *new* core Internet service, namely translating *HyperNames* to lists of IP addresses: in math style: $\mathrm{HN} \Longrightarrow \{\mathrm{IP_i}\}^{\mathrm{i}\in\mathrm{I}}$, with the set I possibly empty in case of content discovery failure. As we did in the Introduction, we choose to present the main features of the CNS service by putting it face-to-face with DNS service.

– The DNS is a "phone book" directory for the Internet. It mainly uses the UDP transport to query other distributed DNS servers to answer client questions like "which IP addresses are associated with `www.google.com`?" The CNS performs a distributed, network aware, content discovery service for the Internet. Similarly to the DNS, the CNS uses the UDP protocol to query other CNS servers to answer client session like *"find some IP addresses that published a content matching the HyperName HN and whose AS is network-close to the AS of the requester"*. Therefore, CNS service also provides information about the hosts that have published a given content;

– The DNS delegates name resolution into *Domain Zones* from the smallest to the biggest zone. Analogously, the CNS delegates content discovery (content name resolution) through *Autonomous Systems* always trying to follow, whether possible, a "reverse cash flow" route in order to suggest to the further content delivery an ordinary "cash flow" route;

– The DNS distributed database is indexed via *Domain Names*. On the other hand, the relations among CNS servers are derived by the relations among the *Autonomous Systems* (i.e., customer-to-provider, provider-to-customer, peering relations). These relations can be derived by using CAIDA's "AS Relationships Dataset" maps (see [6] and Gao's [4]);

```
1.01 on receipt of links(HN,DOWN) from provider do                          receive a query from a ''downhill''
1.02 value = lookupdb(HN);                                                  search HN in the CNS' local data base
1.03 if (value ≠ 0)                                                         some IP publishing HN are found
1.04 then return value to provider;                                         return those IP ''back to the downhill''
1.05 else list = select(α,customerlist);                                    select some customers CNS
1.06      forall cus ∈ list do value = value ∪ send links(HN,DOWN) to cus;  and forward the query downhill through a customer
1.07      return value to provider;                                         return the CNS list (can be empty) ''back to the hill''
```

Fig. 1. `links`: query from downhill continue on α thread downhill

```
2.01 on receipt of links(HN,UP) from peer do                                receive a query from a peer on the ''top of the hill''
2.02 value = lookupdb(HN);                                                  search HN in the CNS' local data base
2.03 if (value ≠ 0)                                                         some IP publishing HN are found
2.04 then return value to peer;                                             return those IP ''back to the top of the hill''
2.05 else list = select(α,customerlist);                                    select some customers CNS
2.06      forall cus ∈ list do value = value ∪ send links(HN,DOWN) to cus;  and forward the query but downhill through a customer
2.07      return value to peer;                                             return the CNS list (can be empty) ''back to the top of the hill''
```

Fig. 2. `links`: query from uphill being on the top of the hill will change on α thread downhill

```
3.01 on receipt of links(HN,UP) from customer do                                receive a query from a ''uphill''
3.02 value = lookupdb(HN);                                                      search HN in the CNS' local data base
3.03 if (value ≠ 0)                                                             some IP publishing HN are found
3.04 then return value to customer;                                             return those IP to ''back to the uphill''
3.05 else list = select(α,customerlist);                                        select some customers CNS
3.06      forall cus ∈ list do value = value ∪ send links(HN,DOWN) to cus;      and forward the query but downhill through a customer
3.07      if (value ≠ 0)                                                        some CNS are suggested
3.08      then return value to customer;                                        return those CNS ''back to the uphill''
3.09      else list = select(γ,peerlist);                                       select some peers CNS
3.10           forall per ∈ list do value = value ∪ send links(HN,UP) to per;   and forward the query uphill through a top of the hill peer
3.11           if (value ≠ 0)                                                   some CNS are suggested
3.12           then return value to customer;                                   return those CNS ''back to the uphill''
3.13           else list = select(β,providerlist);                              select some provider CNS
3.14                forall pro ∈ list do value = value ∪ send links(HN,UP) to pro;  and forward the query uphill through a provider
3.15                return value to customer                                    return the CNS list (can be empty) ''back to the uphill''
```

Fig. 3. `links`: A query from an uphill will continue on three directions: first α-downhill, then γ-downhill, and finally β-uphill

– The DNS queries can be iterative or recursive: the same holds for the CNS, while, as for the DNS, iterative queries are preferred for efficiency reasons.

C. The CNS Hierarchical Topology

To scale up, the content-based distributed database is organized into a *hierarchy of servers* distributed according to the classical Tier-1/Tier-2/Tier-3 AS topology (as described by the AS-to-AS relationship datasets of CAIDA). As well explained by CAIDA, an annotated AS/ISP graph includes relations of kind customer-to-provider (or, symmetrically, provider-to-customer) and of kind peer-to-peer. Customer-to-provider AS relation refers to a relation where the customer ISP pays the provider ISP for transit (the, so called, *flow of money*). Therefore, ASes at lower levels pay ISPs at higher levels in exchange for access to the rest of the Internet. A peering link, instead, connects two ASes who have agreed to exchange traffic on a *quid pro quo* basis. ASes involved in a peering relation exchange traffic only between each other and each other's customers.

Each AS must have at least one CNS server, called *authoritative*, whose database will take into account the association of HN with a list of IPs that have registered a content named by a HN. The authoritative CNS also knows exactly its position in the distributed database, namely (i) the IP addresses of all customers CNS, (ii) the IP addresses of all providers CNS and (iii) the IP addresses of all peer-to-peer CNS: this will allow to dispatch queries along the distributed database.

D. Content Publication

In order to make a content "discoverable", a content must be "published" by some owner or purveyor: this is done simple by sending to the authoritative CNS a message of the shape `send publish(HN) to CNS` where `HN` is the HyperName associated with the given content, `CNS` is the authoritative CNS, and the sender is the owner or the purveyor of the content. Note that the publication in a CNS associate a HN with a principal, and that principal holds the content as an owner or a purveyor (the content being mutable or immutable).

More precisely, suppose a given content C be available by a host belonging to an Autonomous System AS: the host can publish, through the CNS service, the content in the authoritative CNS local database. To do this, at the beginning, the host creates a proper HyperName HN that will be sent as a formal parameter to the authoritative CNS server. Note that the host decides which attribute attach to the HN and whether publish that content as an *owner* or as a *purveyor*. In the first case the publication is done by a simple write in the CNS' database (depending on a local policy, the CNS could ask to republish the content every n seconds). In the second case, the host could be asked to "package" a `.torrent`[2] file and write it in the CNS server database; in the BitTorrent jargon the purveyor play a role of "seed" and it will be asked to republish itself as a purveyor of the content C every m seconds. Further nodes entering the swarm for C will be asked to publish his name in the torrent. In other word, for that content, the CNS server is playing a kind of *network aware BitTorrent tracker*.

E. HyperName Lookup Algorithm (HLA) in a nutshell

As said before, each AS holds an authoritative CNS server, that records the mappings for all the HNs published in the AS. To scale up, the CNS service is organized in a distributed

[2]The BitTorrent is just an example of a distributed file sharing protocol.

hierarchical database, implemented in a hierarchy of servers. Each CNS interacts with the others through a distributed algorithm called *HyperName Lookup Algorithm* (HLA). In a nutshell, when a client asks for a given HN, the following actions are taken (α, β, and γ being AS-specific parameters):
– Step 1. The client first contacts its authoritative server and then searches the HN in the local publications (i.e., in the current and in the peering CNS);
– Step 2. If 1 fail, then the authoritative CNS forwards the query through α-CNS belonging to ASes in downstream, with which we have signed some provider-to-customer agreement;
– Step 3. If the above fails, then the authoritative CNS server forward the query through γ-CNS servers belonging to ASes in peer, with which we have signed some peer-to-peer agreement;
– Step 4. If the above fail, then the authoritative CNS forward the query through β-CNS belonging to ASes in upstream, with which we have signed some customer-to-provider agreement.

F. The `links` pseudocode

A first HLA implementation is showed by the `links`[3] pseudocode in Figures 1, 2, and 3.

a) Start of the HLA: A client sends a query to the authoritative CNS server where the client belongs to, with argument the HyperName HN. In DNS jargon, this query is *recursive* i.e., the client will be blocked until the CNS will answer positively with a result containing a set of addresses $\{\mathrm{IP_i}\}^{\mathrm{i}\in\mathrm{I}}$ associated with HN, or with a search failure.

b) Figure 1: This code refers to the general case when the current CNS receives a `links` message with a HN and a downhill direction from a provider-CNS (line 1.01). First of all, a local lookup is performed (1.02); in case of success, the result value is returned to the sender[4] (1.04); else selects α-customer-CNS (1.05) and sends α-*iterative* `links` queries with the same HN and the same downhill direction (1.06); then collects the result value and send it back to the sender of the first `links` message (1.07).

c) Figure 2: Following the Gao jargon, this code refers to the well-known case of being "on the top of the hill", i.e., receiving a message from uphill and from a peer-to-peer-CNS. Execute the same code as the one of Figure 1, with the following exception: invert the direction from uphill to downhill when sending α-*iterative* `links` queries (2.06).

d) Figure 3: This code refers to the case where a CNS receive a `link` message from uphill from a customer-CNS. Again, following the Gao jargon, when we receive a query from a customer and with an uphill direction the following steps are executed. First of all, a local lookup is performed (line 3.02): in case of success, the result value is returned to the sender (3.04); else select α-customer-CNS[5] (3.05) and send α-iterative `links` queries with the same HN but *inverting the direction* from uphill to downhill (in other words: "repush" downhill the query) (3.06); in case of success, the result value is returned to the sender (3.08); else select γ-peer-CNS (3.09) and send γ-iterative `links` queries with the same HN and the same direction[6] (3.10); in case of success, the result value is returned to the sender (3.12); else select β-provider-CNS (3.13) and send β-iterative `links` queries with the same HN and the same uphill direction (in other words: go uphill only after tried to invert the search downhill but all the queries failed) (3.14); as the last resort of the query, return a success or failure value to the sender. After this code review, the reader has surely observed that:

Note 1: All messages not matching with the above three Figures are simply flushed by the receiving CNS server.

Note 2: In case of α, β, and γ quite small, the number of CNS iterative-messages can be "modest" for a poor resource discovery: therefore, a careful tuning of those parameters is required to find a trade-off between flooding and successful resource discovery. The number of successful queries and the message flooding can be limited (i) by introducing RAM caches in CNS, and (ii) by parsing optional parameters in HN, and (iii) by introducing a "publication lifespan" in CNS publications, and (iv) by introducing TTL in message queries, and (v) by introducing "incentives" to republish in the local AS contents retrieved abroad.

III. Performance Results

This section describes the first evaluation by simulation of the CNS service and of the LogNet Internet architecture extension. In particular, by using real ASes topologies provided by CAIDA [6], we evaluate the sensitivity of the lookup algorithm `links` presented in the previous section with respect to parameters α, β, and γ.

For the experimental evaluation we select an ASes topology provided by CAIDA with 45427 ASes (i.e. a 2013 snapshot).

To this topology we apply a simple classification criterion for identifying Tier-1/Tier-2/Tier-3 [6]. We use this classification to distinguish the behavior of the lookup algorithm for the different type of ASes. A typical case is when the CNS, where the lookup originates, explores its peering neighborhood: a Tier-1 CNS will explores the whole set of the ASes with peering relationships with the CNS, while a Tier-2 (or Tier-3) CNS will performs a random selection on the peering CNSs according to the γ parameter.

In our simulation, we study the performance of lookup algorithm with different set of parameters α, γ, and β and with different distribution of the resource. That is, the resource popularity is the probability that each CNS has a copy of the requested resource. In particular, in our experiment we first evaluate the *Average Lookup Length* (ALL), that is the average number of CNS servers explored during the search phase in case of search failure. Table I summarizes the results obtained for different values of the parameters. Note that these parameters are expressed as a fraction of the AS neighborhood (in downstream/upstream/peering, respectively).

[3]**L**ogNet **I**nternet **N**etwork **K**ey **S**earch.

[4]At the beginning of the search, the sender is just the authoritative-CNS itself, while in the middle of the HLA, the sender is a provider-CNS.

[5]Beware to not choose the customer-CNS that have sent the query.

[6]Beware that successives execution of code in Figure 2 will later invert the direction from uphill to downhill. In short we "repush" downhill the query.

α	γ	β	ALL	α	γ	β	ALL
0.02	0.02	0.001	94.58	0.05	0.05	0.001	164.15
0.02	0.02	0.01	106.7	0.05	0.05	0.01	182.83
0.02	0.02	0.02	116.1	0.05	0.05	0.02	193.67
0.02	0.02	0.03	125.2	0.05	0.05	0.03	204.49
0.02	0.02	0.05	139.4	0.05	0.05	0.05	221.1
0.02	0.02	0.1	161.4	0.05	0.05	0.1	246.29
0.02	0.001	0.02	107.1	0.1	0.05	0.001	248.01
0.02	0.01	0.02	109.2	0.1	0.05	0.01	278.88
0.02	0.02	0.02	116.1	0.1	0.05	0.02	291.57
0.02	0.03	0.02	117.1	0.1	0.05	0.03	305.49
0.02	0.05	0.02	119.1	0.1	0.05	0.05	325.17
0.02	0.1	0.02	120.2	0.1	0.05	0.1	353.56
0.001	0.02	0.02	38.80	0.1	0.001	0.1	3971.4
0.01	0.02	0.02	85.71	0.1	0.01	0.1	3998.8
0.02	0.02	0.02	116.1	0.1	0.02	0.1	4058.6
0.03	0.02	0.02	143.7	0.1	0.03	0.1	4201.7
0.05	0.02	0.02	193.5	0.1	0.05	0.1	4264.6
0.1	0.02	0.02	291.6	0.1	0.1	0.1	4332.8

TABLE I
AVERAGE NUMBER OF CNSS EXPLORED
DURING THE LOOKUP AS FUNCTION OF THE PARAMETERS α, γ, AND β

It can be seen by this first set of experiments the effects of the different values of the parameters on the average number of CNS servers explored during the lookup. It is worth to point out that these parameters are controlled by the CNS (and hence by the corresponding AS) who originates the query and then they can be used to control the fraction of exploited network as function of the relations among the ASes and as function of resource popularity.

Table I presents the ALL computed with a set of parameters that allow to explore a large fraction of the network (e.g. with $\alpha = 1$ and $\beta = 1$ the algorithm explores all the downstream and the upstream neighborhood of each CNS).

Note that the aim of the `links` lookup algorithm is to avoid that the search phase (and as a byproduct the resource exchange) indiscriminately jumps on the different network locations. In the P2P literature this attitude has been called "network un-awareness", but for an AS that subscribes agreements with transit providers this phenomena is no more than an (unjustified and often useless) increasing of transit fees.

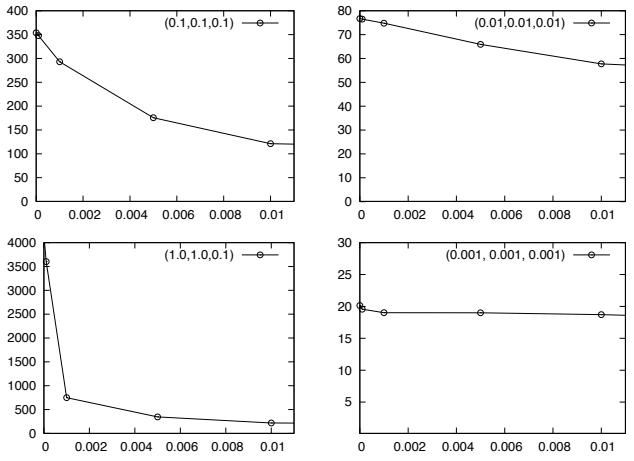

Fig. 4. Average number of CNSs explored by the lookup algorithm as funtion of resource popularity

Figure 4 complements the results presented in Table I by evaluating the impact of the resource popularity and showing the average number of CNSs explored as function of resource popularity (expressed as fraction of CNSs having the resource). Note that in this set of experiments we evaluate the effects on the performance of the lookup algorithm of the different resource popularity expressed in terms of fraction of CNSs having the requested resource. The figure shows four plots corresponding to four different sets of parameters. We can see how increasing the resource popularity improves the performance of the lookup algorithm and this effects is stronger when the three parameters are such that the number of exploited CNSs is higher. Note that this enables the exploitation of tradeoffs between resource popularity vs number of explored CNSs.

Although the analysis we present in this section does not account for the dynamic evolution of the resource popularity (i.e., we are assuming here that the resource popularity does not change during the lookup phase), the insight it provides can be used by the CNSs to explore a wide set of parameters vs the resource popularity. In particular, each CNS can modulate the costs of the lookup phase in terms of number of explored CNSs (and hence of the distance in terms of AS hops). In other words, the lookup algorithm throughout the parameters modulates the CNS's network awareness. Furthermore, since each CNS is aware of its connectivity relations and of the transit costs related with these relations, the lookup parameters can be seen as an (indirect) way to optimize the AS's transit costs.

IV. FURTHER DEVELOPMENTS

This final section shortly discuss related works, HLA optimizations, additional services provided by CNS, issues related to mobility, nomadism and security issues. As usual the interested reader can have a look on the web appendix.

A. LogNet vs. DONA

We are respectfully and genuinely indebted with all the ICN literature. Our objective was not to set up yet another proposal but try to get the best of every architecture and proposed protocol, and having in mind that we wanted to build a conservative extension of the actual IP hourglass architecture. We were also greatly inspired to a logical network architecture by the first author, called ARIGATONI, featuring resource discovery and virtual intermittence protocols [9]–[11]. CAIDA maps were also useful, and DNS name resolution lookup was also a source of inspiration.

We think that the proposal which merits a precise comparison with our is DONA [2] (always in **Q&A** style).

Q: *Does LogNet have some analogies with the DONA proposal, and in particular the role of CNS vs. the DONA's Resolution Handlers (RH)?*

A: DONA relies on flat, machine readable, persistent names, of the shape `P:L`, where `P` is the cryptographic hash of the "principals public key" and `L` is a label chosen by the principal. LogNet relies on HN having a more elastic syntax, made by a human readable content name concatenated by a number of (optional) parameters helping to identify a content,

and, possibly, its ownership, its integrity, its hosting, and its attribute-list. DONA's labels and the LogNet's content name could be the same, but the elastic structure of HN is richer and flexible than the one of DONA, being `P` one possible optional parameter in the LogNet's HN. Each CNS server memorize *only* pointers to local publications, while the DONA's RH maintains a registration table that maps a name to both a next-hop RH and the distance to the copy (in terms of the number of RH hops), or directly holds the resource. DONA's RH and LogNet's CNS hierarchies cannot be overlapped, since the latter overlaps by definition the AS-topology (one CNS per AS): finally the `links` lookup algorithm promote location awareness and valley free routing, while the DONA's `FIND` lookup algorithm implements routing to a closest copy.

B. HLA Optimizations

The curious reader can have a look on the web appendix for a list of possible optimizations of the HLA that will be subject of further study. The more promising ones are:

– opt 1. Adding caches (maintained in RAM) to all the CNS databases could leverage the number of message exchanges between CNS and can be useful in case of *"flash crowds"*.

– opt 2. Processing HN's optional parameters could improve the pattern matching algorithm hidden in the HLA algorithm. Succinctly: `tags` can modify the choice of α, β, γ CNS to which forward the query lookup; `hosts` allows to fetch the content name directly from one of the hostnames in the list; `fing_cont` allows to improve the HLA in case of immutable contents; `fing_princ` allows to improve the HLA in case of mutable contents of a unique owner.

– opt 3. Introducing a "lifespan" in CNS publications allows to better organize CNS databases and to improve caches performance.

– opt 4. Introducing a TTL in `links` messages permits to flush messages whose counter has elapsed and limits flooding in the HLA.

– opt 5. Adding "incentives" to locally published contents introduce some good habits to diffuse immutable contents in the CNS's distributed hierarchical database.

C. Additional services that could be provided by a CNS

The primary mission of a CNS server is the one of translating HN into a set of IP addresses. Nevertheless, CNSs could have other missions and in particular *Content Aggregation* and *Load Distribution*. In a nutshell:

– Content Aggregation concerns the aggregation of publication of two *semantically equal* contents (i.e. having the same `fing_cont`) with two *syntactically different* HN. This is the case where the CNS future *Aggregation Algorithm* (`a2`) will try to merge some entries in order to keep the content table most faithful as possible (so dealing with structural/lexical heterogeneity).

– Load Distribution of replicated copies of a single content in different CNS servers. If CNS tables maps a HN into a lists of IP sets, then the CNS can respond with the entire list of nodes, or it can rotates the ordering of the addresses within each reply. As such, IP rotation performed by CNS can distributes among multiple purveyors.

D. Mobility, Nomadism, and Security

As any decent Information Centric Network proposal, LogNet should take into account mobility and nomadism and security issue. We shortly deal with those three points leaving the interested reader to browse the web appendix.

Mobility. The difficulty resulting of dealing with mobility could arise especially in case the owner/purveyor is a *mobile host*. The symmetric case where the client is a mobile node can, without loss of generality, be considered out of the scope of the discovery service.

Nomadism. The publisher is a mobile node: we must distinguish the case of immutable or mutable contents. Immutable contents can be published proviso good garanties of "permanence" in the AS of the purveyor, while mutable contents can be published under the presence in the HN of the logical attribute `fing_princ` and when the logical attribute `hosts` contains only one symbolic name or only one IP.

Security. DNS history teaches us that, up to date, there has been no significant DNS attacks that has successfully impeded the distributed DNS service (i) because DNS servers are machines managed and "protected" by system administrators, and (ii) because the DNS protocol pushes lookup always "below" the hierarchical database, minimizing the "uphill ascents", and (iii) because of making use of well-known techniques of caching. May the CNS service have the same chance?

REFERENCES

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. of CONEXT*. ACM, 2009, pp. 1–12.

[2] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 181–192, 2007.

[3] L. Liquori and M. Sereno, "Web appendix of the paper," https://www.dropbox.com/s/21i4s4gwxwlaj3s/1-wapp.pdf?dl=0 See also tech rep. on https://hal.inria.fr/hal-01323974 .

[4] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 733–745, 2001.

[5] G. D. Battista, M. Patrignani, and M. Pizzonia, "Computing the types of the relationships between autonomous systems," in *Proc. of INFOCOM*. IEEE, 2003, pp. 156–165.

[6] CAIDA, "Center for Applied Internet Data Analysis: AS relationship," http://www.caida.org/data/as-relationships/, 2016.

[7] S. Y. Qiu, P. D. McDaniel, and F. Monrose, "Toward valley-free interdomain routing," in *Proc. of ICC*. IEEE, 2007, pp. 2009 – 2016.

[8] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19, no. 1, pp. 1–16, 2007.

[9] R. Chand, M. Cosnard, and L. Liquori, "Powerful Resource Discovery for Arigatoni Overlay Network," *Future Generation Computer Systems*, vol. 24, no. 1, pp. 31–48, 2008.

[10] D. Benza, M. Cosnard, L. Liquori, and M. Vesin, "Arigatoni: A Simple Programmable Overlay Network," in *Proc. of JVA, IEEE,* pp. 82-91, 2006.

[11] L. Liquori and M. Cosnard, "Logical Networks: Towards Foundations for Programmable Overlay Networks and Overlay Computing Systems," in *Proc. of TGC*, ser. LNCS, vol. 4912. Springer, 2007, pp. 90–107.

LogNet: Extending Internet with a Network Aware Discovery Service

[Web Appendix]

Luigi Liquori
Université Côte d'Azur
Inria Sophia Antipolis Méditerranée, France
Email: Luigi.Liquori@inria.fr

Matteo Sereno
Università di Torino, Italy
Email: Matteo.Sereno@unito.it

I. Bonus Content Name System

In the last decade, Information Centric Networking research has produced, a lot of very interesting proposals. These proposals (or some of them) will be the basis for future network architectures. On the other hand, in this paper we will use some of the ideas developed by the ICN research venue to propose a content discovery service that can be implemented in the current Internet and that (hopefully) can alleviate some of the problems in the content distribution process.

A. Clean-state Design vs. Evolution of the Existing Network Architecture

While it is always exciting to conceive a new network starting from new concepts and from a clean-state design, network's history teaches us that the Internet infrastructure and its protocol suite have little changed especially at the lower level of the OSI stack; this is, quite obviously, because of strong backward-compatibility needs, and because of the tremendous expansion of the Internet phenomenon.

This paper supports the "evolutive" research line. Let us consider a programming language analogy: machine code was used at the very beginning of computers science by humans as the only (low level) programming language but later still employed by compilers as a target language to which map high level programming languages. In particular, we believe that the current TCP/IP-based transport protocol could be considered as a target protocol suite to efficiently map more sophisticated protocols and services. It is authors' believe that a lot of the above cited ICN proposals, including the one presented in this paper, could (and should) be deployed by a small, surgical, conservative extension on the current TCP/IP hourglass Internet infrastructure. In terms of protocol stack the CNS service can be located in the IP hourglass at the same level of the DNS, as Figure 1 pictorially shows.

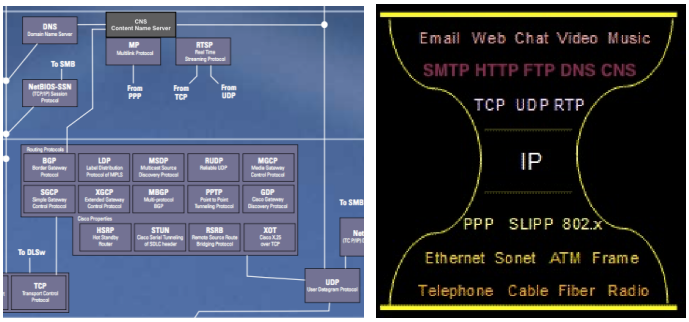

Fig. 1. Zoom of network protocols "close" to CNS and new hourglass IP model

B. Mobility, Nomadism and Security in CNS

Mobility: Since traffic from wireless and mobile devices will probably exceed traffic from wired devices by 2016, we could expect that most contents could be requested and delivered by both wireless and mobile devices. It is well known that *"wireless and mobile devices may easily switch networks, changing their IP address and thus introducing new communication modalities based on intermittent and, possibly, opportunistic connectivity"* [1]. Since the LogNet architecture consists of adding a content discovery service to the current Internet, it follows that the difficulty resulting of dealing with mobility could arise especially in case the owner/purveyor is a *mobile host*. The symmetric case where the client is a mobile node can, without loss of generality, be considered out of the scope of the discovery service.

Nomadism: The publisher is a mobile node. In this case the mobile node wants to publish a content: two cases can happen according to the (im)mutability of the content:

- *Immutable:* (this case being surely the most common of the two). The authoritative CNS related to the mobile ISP could accept the publication of an immutable content by a mobile user with the proviso of (i) recording the identity of the user, via e.g. the MAC address of the mobile device (or another identifier of the mobile node), and (ii) asking to the mobile user to re-publish the content more frequently than a fixed device, and (iii) possibly "blacklisting" a mobile device that "publish and disappear" too fast or too often.
- *Mutable:* this case is less common but quite challenging since it deal with the possibility to keep an identity also in case the user is navigating through different mobile networks. The authoritative CNS related to the mobile ISP could accept the publication of a mutable content

if and only if the logical attribute `fing_princ` is present and the logical attribute `hosts` contains only one symbolic name or only one IP.

Security: DNS history teaches us that, up to date, there has been no significant DNS attacks that has successfully impeded the distributed DNS service: the secret of this success story was especially (i) because DNS servers are machines managed and "protected" by system administrators, and (ii) because the DNS protocol pushes lookup always "below" the hierarchical database, minimizing the "uphill ascents", and (iii) because of making use of well-known techniques of caching. Do CNS could have the same chance? We honestly don't know, a more deep study of this protocol is undergo: nevertheless the following indices make us to think positive: (a) the 70K+ CNS servers could be managed by AS system administrators and (b) the HLA always pushes routing first downhill the customer-CNS distributed database, and, only in case of failure, uphill through a peer-CNS or a provider-CNS. Anyway, in its current essential formulation, the CNS service is not vaccinated by well-known attacks like, DDoS bandwidth-flooding attack, or man in the middle attack, or poisoning attack, or spoofing an IP of a node below an authoritative CNS.

II. Bonus: HLA Optimizations

A. HLA Optimization 1: Adding Cache in CNS

In perfect analogy with DNS, a CNS could put in a cache the result of a successful query lookup giving positive results not in the current AS. The positive effect of a small caches (maintained in RAM) applied to all the CNS databases could leverage the number of message exchanges between CNS. Because the presence of a CNS mapping is no more permanent, CNS servers could also discharge cached records after a period of time to be fixed in the `links` codebase. Caches are shown very useful in case of HN that catches the interest of a large number of clients, and get unexpected overloading of CNS-traffic (sometime called *"flash crowd"*).

B. HLA Optimization 2: Processing HN's Optional Parameters

As said before, an HN can have the following form:

```
[fing_princ:][fing_cont:][hosts:][tags:]cont_name
```

Until now, all optional parameters (logical attributes) were unused in the pattern matching algorithm hidden in the HLA algorithm. As such, a possible optimization should concern how to process the four logical attributes.

- (`tags`) A list of tags can be attached to an HN in order to choose α, β, γ CNS to which forward the query lookup. The purpose is to limit the search space and improve the success rate. As shown in the `links` pseudocode, the choice of α, β, and γ is fixed once for all, but a better use of the list of tags can be done as follows: suppose that each CNS have a data structure called Tag_rate associating to each tag `t` a triple of probabilities $(prob^{\mathtt{t}}_{\alpha}, prob^{\mathtt{t}}_{\beta}, prob^{\mathtt{t}}_{\gamma})$. The Tag_rate structure is updated each time a consumer-CNS, a peer-CNS, or a producer-CNS reply successfully for an HN tagged with `t`. The flooding parameters in the `links` instructions (1.05,2.05,3.05,3.09,3.12) will be adjusted following the previous tag success rate. In few words, the lookup history can greatly customize and improve the CNS routing.
- (`hosts`) When an hostname list prefixes an HN, this means that the content name must be directly retrieved from one of the hostnames in the list. In this case, the local CNS just perform a DNS query to transform the given hostname into a single IP address and return, leaving the content transfer outside the scope of the CNS service.
- (`fing_cont`) This logical attribute allows to improve the HLA in case of immutable contents: when a content's fingerprint prefixes an HN, this means that the integrity of the content to be retrieved can immediately be verified as soon as we retrieve the content itself.
- (`fing_princ`) This logical attribute allows to improve the HLA in case of mutable contents of a unique owner: when a principal's fingerprint prefixes an HN, this means that when the client receive a content together with the public key of the owner, the identity of the latter can be immediately verified as soon as we retrieve the content itself.

C. HLA Optimization 3: Introducing a "Lifespan" in CNS publications

This simple optimization allows to better organizes CNS databases and to improve caches performance: it states that each publication in an authoritative CNS has a lifespan: after the end of the lifespan, either the publisher re-publish the content in the CNS, or the record is simply dropped out from the CNS.

D. HLA Optimization 4: Introducing a TTL in `links` messages

This simple optimization allows to limits the lifetime of lookup messages. A "Time to Live" (TTL) counter attached to each `links` message permits to "flush" messages whose counter has elapsed. This also limits message flooding in the HLA.

E. HLA Optimization 5: Adding "Incentives" to Locally Publish Messages

This simple optimization introduce some good habits to diffuse immutable contents in the distributed hierarchical database of CNS: it states that every client using the CNS discovery service, should get some incentives to "locally republish" some contents in case the discovery service answer returning a pointer to a content in another AS. A "tit for tat" strategy could be installed between clients – looking for contents – and purveyors – distributing the contents – were the CNS should play a special (business? content reputation?) role being in the middle of the above two actors.

III. Bonus: Additional services provided by a CNS

The primary mission of a CNS is the one of translating HN into a set of IP addresses. Nevertheless, CNS could have other missions and in particular *Content Aggregation* and *Load Distribution*.

A. Content Aggregation in CNS

This service concerns the aggregation of publication of two *semantically equal* contents (i.e. having the same `fing_cont`) with two *syntactically different* HN. More precisely, every time a purveyor publish an immutable content with a given HN2, the authoritative CNS can verify that the same content is not already published with a similar but equationally different HN1[1]. We explain the problem in short: if `fing_cont` is omitted in one HN, then there is no way to formally check equality between two contents having potentially different and "incompatible" tag lists. Note that the concept of "tag-incompatible" is a semantic one and not just syntactic. Indeed slightly different content names, differing, e.g., for tiny typographic errors and incompatible host name lists could refers to the *same real content*. This is the case where the CNS *Aggregation Algorithm* (`a2`) will try to merge some entries in order to keep the content table most faithful as possible (so dealing with *structural/lexical heterogeneity*)

Just to give an example of the many potential aggregations, let the following two different entries

```
HN1 = fing_cont:hosts1:tags1:cont_name1
HN2 = fing_cont:hosts2:tags2:cont_name2
```

be published in some authoritative CNS. The acute reader can see that HN1 and HN2 differs in content names and in all logical attributes but the digital signature of the content which is the same. A content aggregation will rewrite the previous two entries and substitute with the following ones:

```
HN1 = linksto HN3
HN2 = linksto HN3
HN3 = fing_cont:hosts1,hosts2:tags1,tags2:
                        cont_name1|cont_name2
```

where the symbol "`,`" denotes list concatenation and the symbol "`|`" denotes an "or" operator that allow to match both content names in pattern matching.

Note that *Data deduplication* [2] is an emerging technology that introduces reduction of storage utilization: the study of `a2` is out of the scope of this paper, and will be described in a future work.

B. Load distribution

As DNS does, CNS can perform load distribution among replicated copies of a single content. If CNS tables maps an HN into a *lists of IP sets*, then the CNS can respond with the entire list of nodes, or it can "rotates" the ordering of the addresses within each reply. As such, IP rotation performed by CNS can distributes among multiple purveyors.

[1]The "smarphone-addicted" readers have often experienced this in merging different contact lists.

C. Related Work

Without any doubt, we are respectfully and genuinely indebted with all the literature cited in this appendix: first of all the generous tutorials and internet draft, which allowed to warm us; then all the nice ICN proposals we have read. Our objective was not to set up yet another proposal but try to get the best of every architecture and proposed protocol, and having in mind that we wanted to build a conservative extension of the actual IP hourglass architecture. Finally, we were also inspired to a logical network architecture, called ARIGATONI, we developed some year ago, featuring resource discovery and virtual intermittence protocols [3]–[5]. CAIDA maps were also more than useful, and DNS protocol was also source of great inspiration.

References

[1] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *COMMUNICATIONS SURVEYS AND TUTORIALS*, vol. 16, no. 2, pp. 1024–1049, 2014.

[2] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19, no. 1, pp. 1–16, 2007.

[3] R. Chand, M. Cosnard, and L. Liquori, "Powerful Resource Discovery for Arigatoni Overlay Network," *Future Generation Computer Systems*, vol. 24, no. 1, pp. 31–48, 2008.

[4] D. Benza, M. Cosnard, L. Liquori, and M. Vesin, "Arigatoni: A Simple Programmable Overlay Network," in *Proc. of JVA, IEEE,* pp. 82-91, 2006.

[5] L. Liquori and M. Cosnard, "Logical Networks: Towards Foundations for Programmable Overlay Networks and Overlay Computing Systems," in *Proc. of TGC*, ser. LNCS, vol. 4912. Springer, 2007, pp. 90–107.

[6] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. of CoNEXT*. ACM, 2009, pp. 1–12.

[7] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 181–192, 2007.

[8] G. D. Battista, M. Patrignani, and M. Pizzonia, "Computing the types of the relationships between autonomous systems," in *Proc. of INFOCOM*. IEEE, 2003, pp. 156–165.

[9] S. Y. Qiu, P. D. McDaniel, and F. Monrose, "Toward valley-free interdomain routing," in *Proc. of ICC*. IEEE, 2007, pp. 2009 – 2016.

[10] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 733–745, 2001.

[11] L. Liquori and M. Sereno, "Web appendix of the paper," https://www.dropbox.com/s/21i4s4gwxwlaj3s/1-wapp.pdf?dl=0 See also tech rep. on https://hal.inria.fr/hal-01323974 .

[12] CAIDA, "Center for Applied Internet Data Analysis: AS relationship," http://www.caida.org/data/as-relationships/, 2016.

[13] M. Gritter and D. R. Cheriton, "An architecture for content routing support in the internet," in *Proc. of USITS*. USENIX Association, 2001, pp. Vol 3, 4–4.

[14] P. Jokela, A. Zahemszky, C. Esteve Rothenberg, S. Arianfar, and P. Nikander, "Lipsin: Line speed publish/subscribe inter-networking," in *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, ser. SIGCOMM '09. ACM, 2009, pp. 195–206.

[15] CAIDA, "CAIDA AS relationship," http://data.caida.org/datasets/as-relationships/, 2016.

[16] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker, "Rofl: Routing on flat labels," in *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '06. ACM, 2006, pp. 363–374. [Online]. Available: http://doi.acm.org/10.1145/1159913.1159955

[17] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. of CONEXT*. ACM, 2009, pp. 1–12.

[18] D. Smetters and V. Jacobson, "Securing network content," Palo Alto Research Center, Tech. Rep., 2009.
[19] M. Clark, "Post congress tristesse," in *TeX90 Conference Proceedings*. TeX Users Group, March 1991, pp. 84–89.
[20] A. Detti, N. Blefari-Melazzi, S. Salsano, and M. Pomposini, "CONET: a content centric inter-networking architecture," in *2011 ACM SIGCOMM Workshop on Information-Centric Networking, ICN*, 2011, pp. 50–55.
[21] D. Lagutin, K. Visala, and S. Tarkoma, "Publish/subscribe for internet: PSIRP perspective," in *Towards the Future Internet - Emerging Trends from European Research*, 2010, pp. 75–84.
[22] A. Venkataramani, J. F. Kurose, D. Raychaudhuri, K. Nagaraja, M. Mao, and S. Banerjee, "Mobilityfirst: a mobility-centric and trustworthy internet architecture," *Computer Communication Review*, vol. 44, no. 3, pp. 74–80, 2014.
[23] M. Cosnard, L. Liquori, and R. Chand, "Virtual Organizations in Arigatoni," in *Proceedings of the Second International Workshop on Developments in Computational Models (DCM 2006)*, ser. Electronic Notes in Theoretical Computer Science, vol. 171- issue 3. Elsevier, 2006, pp. 55–75.
[24] M. Cosnard and L. Liquori, "Weaving Arigatoni with a graph topology," in *1st International Conference on Advanced Engineering Computing and Applications in Sciences ADVCOMP 2007*, C. S. Press, Ed., 2007, pp. 55 – 59.
[25] L. Liquori, D. Borsetti, C. Casetti, and C.-F. Chiasserini, "An Overlay Architecture for Vehicular Networks," in *Proc. of NETWORKING 2008. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet 7th International IFIP-TC6 Networking Conference Singapore,* , ser. Lecture Notes in Computer Science, vol. 4982. Springer Verlag, 2008, pp. 60–71.